SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

August 2020, Vol. 63 182104:1–182104:14
https://doi.org/10.1007/s11432-019-2771-0

Important sampling based active learning for imbalance classification

Xinyue WANG[1,2], Bo LIU[3], Siyu CAO[1], Liping JING[1,2]* & Jian YU[1,2]

[1]*School of Computer and Information Technology, Beijing Jiaotong University, Beijing* 100044*, China;*
[2]*Beijing Key Lab of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing* 100044*, China;*
[3]*College of Information Science and Technology, Hebei Agricultural University, Baoding* 071001*, China*

Received 26 September 2019/Revised 11 December 2019/Accepted 19 January 2020/Published online 7 July 2020

Abstract Imbalance in data distribution hinders the learning performance of classifiers. To solve this problem, a popular type of methods is based on sampling (including oversampling for minority class and undersampling for majority class) so that the imbalanced data becomes relatively balanced data. However, they usually focus on one sampling technique, oversampling or undersampling. Such strategy makes the existing methods suffer from the large imbalance ratio (the majority instances size over the minority instances size). In this paper, an active learning framework is proposed to deal with imbalanced data by alternative performing important sampling (ALIS), which consists of selecting important majority-class instances and generating informative minority-class instances. In ALIS, two important sampling strategies affect each other so that the selected majority-class instances provide much clearer information in the next oversampling process, meanwhile the generated minority-class instances provide much more sufficient information for the next undersampling procedure. Extensive experiments have been conducted on real world datasets with a large range of imbalance ratio to verify ALIS. The experimental results demonstrate the superiority of ALIS in terms of several well-known evaluation metrics by comparing with the state-of-the-art methods.

Keywords imbalance classification, important sampling, active learning, oversampling, undersampling

Citation Wang X Y, Liu B, Cao S Y, et al. Important sampling based active learning for imbalance classification. Sci China Inf Sci, 2020, 63(8): 182104, https://doi.org/10.1007/s11432-019-2771-0

1 Introduction

The imbalanced data is ubiquitous and inevitable, which has been a critical issue in community of machine learning. Taking binary classification into consideration, imbalance means one class (majority or negative class) outnumbers the other class (minority or positive class). The imbalance ratio between classes may range from ten to hundred or even larger in real applications. In extreme multi-label learning [1], each tail class only covers few instances while head class contains a large amount of instances. In dense object detection, background samples extremely outnumber foreground (object) samples [2]. Such imbalanced data brings two challenges for classification task: lack of minority class instances and noisy information in majority class. The former is a terrible issue because the learning model cannot be sufficiently trained to determine the hidden pattern for minority class. The second issue will result in a decision boundary far away from the ideal one owing to the effect of noisy data. Thus, it is necessary to leverage the minority class and remove redundant instances from the majority class.

To date, sampling-based technique has been a direct and simple way to relieve the imbalance problem with the aid of increasing the minority class instances (oversampling) or decreasing the majority class

* Corresponding author (email: lpjing@bjtu.edu.cn)

instances (undersampling). To boost the generalization ability for imbalance classification, the joint methods are proposed by taking advantage of both undersampling and oversampling [3, 4]. Although hybrid sampling methods utilize these two sampling strategies simultaneously, they still suffer from more or less drawbacks of these two sampling strategies, which are performed separately. Active learning is an alternative strategy to handle the imbalance classification task. Its main idea is to add informative instances during active learning process [5, 6]. Ensemble learning and cost-sensitive learning have also been utilized to improve the performance of imbalanced data classifier. Even though these methods obtain promising results on imbalance classification, ensemble methods suffer from how to generate the accurate ensemble components and design proper fusion function to combine the weak classifiers. Similarly, it is expensive to manually tune the cost parameter for cost-sensitive learning methods even with the aid of domain knowledge [7].

Motivated by the above-mentioned observations, we propose a new active learning framework by alternative performing important sampling (ALIS) for imbalance classification. Its main idea is to iteratively select important majority-class instances and generate informative minority-class instances. The other important point is that the proposed active learning's querying strategy has ability to yield a balanced training set in each learning stage. For the important undersampling in ALIS, a new active selection criterion is proposed which captures the first- and second-statistics of the selected majority class set. In this case, diverse and informative majority class instances will be added to the training set. For the important oversampling in ALIS, a new generator is proposed which captures the distribution of informative sampling seeds in the minority class. In this case, each seed is considered as a center and the corresponding bandwidth can be adaptively determined which has ability to introduce data perturbation for robust learning. Following this distribution, we can make sure the synthetic samples are more diverse than random replication and linear interpolation. More specially, all minority-class instances and partial majority-class instances (with similar size of minority class) are taken as the input of active learning to build the classifier in each iteration. Based on the current boundary, the important majority-class instances will be selected from the pool if the proposed active selection criterion is satisfied. With the increasing of the selected majority-class instances in each iteration of active learning, the synthetic and informative minority-class instances will be simultaneously generated by estimating the density distribution of the minority-class instances around the current boundary. Thus, the proposed ALIS is definitely from the existing methods which apply sampling (oversampling minority class or undersampling majority class) once and build the classifier.

The main contributions of our work can be summarized as follows.

- To the best of our knowledge, this is the first imbalance classification work with simultaneously undersampling on majority class and oversampling on minority class under the active learning framework.
- A new undersampling strategy is proposed according to the distribution of boundary majority-class instances, which has ability to select important majority-class from pool for the subsequent learning process.
- A new oversampling strategy is proposed to select the borderline minority-class instance as sampling seeds and adaptively generate informative minority-class instances, which make sure the training set is balanced in each learning stage.
- A series of experiments have been conducted on real-world datasets with large range of imbalance ratio to demonstrate the superiority of ALIS by comparing with the state-of-the-art methods.

The rest of paper is organized as follows. Section 2 provides the related work. The proposed ALIS framework is described in Section 3. Section 4 reports the experiments in detail and discusses the experimental results. Finally, a brief conclusion is drawn in Section 5.

There is an increasing interest in imbalanced data classification task. Till now, lots of methods have been proposed, which can be roughly divided into three main categories: cost sensitive-based, sampling-based and active learning-based methods. Among them, cost sensitive-based methods [8–11] are designed by introducing class-dependent or example-dependent cost to a traditional classification model. But it is hard to predefine the proper values for the cost variables.

2 Related work

2.1 Sampling-based methods

The main goal of sampling-based methods is to make the imbalanced data become balanced via oversampling or undersampling techniques, and then traditional classification approaches can be applied. More specifically, oversampling aims at increasing minority-class instances, while undersampling tries to decrease the number of majority-class instances.

The popular oversampling strategies mainly contain replicating the existing minority-class instances and generating new instances for minority class. In the former strategy, each minority-class instance may be replicated several times, which may cause overlapping and further destroy the classification performance. Thus, most researchers focused on the latter strategy and proposed a series of methods to create the synthetic minority-class instances. The typical one, SMOTE [12], adopts linear interpolation to generate new points between the selected minority-class instances which are called as sampling seeds.

The main factor affecting the performance of SMOTE-type oversampling methods is sampling seeds selection. To identify the informative sampling seeds, the k-nearest neighbor technique is widely used to weigh the minority-class instances and detect the proper seeds according to their importance [13, 14]. Meanwhile, the data distribution around the sampling seeds is used to determine the amount of new generated instances [15]. To consider the effect of majority class, Barua et al. [16] presented a majority weighted minority oversampling technique (MWMOTE) to determine informative minority sampling seeds, and generate new instances by considering the closeness and density of each seed.

Although oversampling methods obtain promising result in imbalanced data classification, they usually suffer from overfitting problem and noisy information. To remove noisy samples, an undersampling technique is introduced to handle majority class, where they make use of clustering, cleaning strategy, etc. [17, 18]. However, it is hard to distinguish noisy instances from the whole dataset, which may result in discarding useful information. Therefore, researchers take advantage of undersampling and oversampling together via ensemble learning or active learning.

In ensemble learning-based imbalanced classification, bagging and boosting are the popular techniques. Bagging methods fuse several weak classifiers with equal weights, where each classifier is built on a subset of training data [3, 4, 19]. Boosting methods incrementally build a stronger classifier by enforcing a heavy penalty on the instances that the previous classifier mis-predicted [20, 21]. Unfortunately, there are still some open problems in ensemble learning, such as how to guarantee the diversity and accuracy for ensemble components, how to make sure that the sampling results keep the structure information of original data.

2.2 Active learning-based methods

Active learning involves designing a selection strategy to select informative instances and ask an oracle to label them, which helps reduce labeling effort in classification problems [22–25]. Recently, it has been introduced in imbalanced classification, which is leveraged to obtain the informative instances when there are sufficient labeled data [5, 26]. Its main idea is to incrementally add instances during learning process, which makes classification performance better by adding informative instances from both classes or avoids the imbalance ratio becoming larger by adding minority class instances. The key issue of active learning-based imbalanced classification is designing the criterion to select proper instances from the pool.

In imbalance classification, the separating hyperplane is skewed towards the minority class, which would finally yield a suboptimal model [27]. To adjust learning model by informative instances, the active selection criteria are designed by taking the character of the model into consideration, which has been integrated with as support vector machine (SVM) [26], Bayes classifier [28] and so on. Thus, in the case of applying SVM to deal with imbalance classification under active learning, the instances closest to the current hyperplane are selected to update the model in the next round of training process. For naive Bayes based active learning method, the training data set is expended by the examples that can maximize information gain of the Bayes classifier before and after adding them. However, this type of

Table 1 Notations in ALIS framework

Notation	Description
$\mathcal{P}$	Original positive class set
n^+	The number of positive instances
$\mathcal{N}$	Original negative class set
n^-	The number of negative instances
$\mathcal{D}$	Original training set and $\mathcal{D} = \mathcal{P} \cup \mathcal{N}$
n	The number of training instances and $n = n^+ + n^-$
$\mathcal{P}^j_{\text{active}}$	Generated synthetic positive class set in the jth iteration
$\mathcal{N}^j_{\text{active}}$	Selected negative class set in the jth iteration
$\mathcal{N}_{\text{pool}}$	Remaining negative class set after active selection
$\mathcal{P}_{\text{active}}$	Generated synthetic positive class set, $\mathcal{P}_{\text{active}} = \bigcup_j \mathcal{P}^j_{\text{active}}$
$\mathcal{N}_{\text{active}}$	Selected negative class set, $\mathcal{N}_{\text{active}} = \bigcup_j \mathcal{N}^j_{\text{active}}$
ω	A linear predictor
f	A linear model
λ_1	The trade-off parameter for controlling the margin variance
λ_2	The trade-off parameter for controlling the margin mean

criteria may be not effective to make up for the imbalance between classes. Thus, to reduce the imbalance ratio by adding more minority class instances, the specialist criteria are usually incorporated a preference for selecting minority class instances [29, 30].

Active learning aims at employing informative training instances iteratively, which has been widely used in imbalance learning [5]. But, the active selection procedure usually fails to expand training set with important instances as well as balance the unequal distribution between classes. Thus, it is essential to generalize existing active learning strategy for imbalance classification problem.

3 Important sampling based active learning

Let $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\,|\boldsymbol{x}_i \in \mathbb{R}^m, y_i \in \{+1, -1\}, i = 1, \ldots n\}$ denote an imbalanced binary training set with n points. In $\mathcal{D}$, $\boldsymbol{x}_i$ indicates the ith instance vector in m-dimension space, y_i is its target value, where $y_i = +1$ means the instance belongs to the positive class, otherwise it belongs to the negative class. Conveniently, majority (negative) class and minority (positive) class are marked as $\mathcal{N}$ and $\mathcal{P}$ respectively, and n^+ and n^- are the amount of $\mathcal{N}$ and $\mathcal{P}$ respectively, thus $\mathcal{D} = \mathcal{P} \cup \mathcal{N}$ and $n = n^+ + n^-$. The notations used in this study are listed in Table 1.

3.1 ALIS framework

The main goal of classifier is to determine the boundary between minority class and majority class by analyzing the training data. The instances closest to the hyperplane are more informative than the ones far from the decision boundary. To reach this goal, we propose an ALIS. It contains two main parts, one for important undersampling to select the important majority instances by considering the distribution of majority class, and the other for important oversampling to generate new important minority instances by estimating the distribution of boundary minority instances.

The proposed ALIS framework is shown in Figure 1. To sufficiently mine latent information of the rare positive instances, all positive instances $\mathcal{P}$ are used at the beginning of active learning process, i.e., $\mathcal{P}^0_{\text{active}} = \mathcal{P}$. To avoid imbalanced data set, only partial negative instances are randomly selected to form the initial negative set, denoted as $\mathcal{N}^0_{\text{active}}$; especially, we set $|\mathcal{N}^0_{\text{active}}| = |\mathcal{P}^0_{\text{active}}| = |\mathcal{P}| = n^+$. Next, any traditional classifier can be trained on this balanced dataset $\{\mathcal{P}^0_{\text{active}}, \mathcal{N}^0_{\text{active}}\}$. For the jth iteration of active learning, ALIS expands the training set with informative negative set $\mathcal{N}^j_{\text{active}}$ which are selected from the pool $\mathcal{N}^{j-1}_{\text{pool}}$, where $\mathcal{N}^{j-1}_{\text{pool}}$ is the left negative instances set after selecting the important negative instances in the previous $j-1$ rounds. Meanwhile, ALIS generates new informative positive instance $\mathcal{P}^j_{\text{active}}$ to balance the training set. In this case, the selected representative negative instances

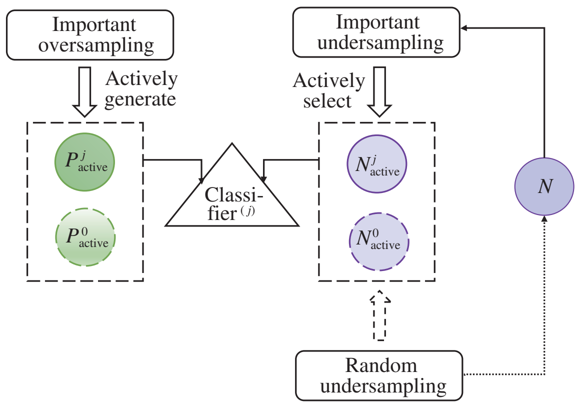

Figure 1 (Color online) The schematic diagram of ALIS framework. The classifier is initially trained by all positive instances P^0_{active} and equal amount of random negative instances N^0_{active}, and is updated iteratively according to new selected negative points or new generated positive points.

and generated informative positive instances are expected to improve the active learning performance and build more effective classifier for this iteration.

Like any active learning method, we have to design the stopping condition (stop criterion) to terminate the training process. Recall that our goal is to establish a stable and effective classifier, thus, we take the predication performance of non-selected negative training instances as the criterion. Once the difference between the performance in the Tth iteration and that in the $(T-1)$th iteration is smaller than a threshold, we will stop the active learning process. In this case, the final classifier is built with the original positive instance $\mathcal{P}$, initial randomly selected negative instance $\mathcal{N}^0_{\text{active}}$, oversampled positive set $\mathcal{P}_{\text{active}} = \bigcup_{j=1}^{T} \mathcal{P}^j_{\text{active}}$, and undersampled negative set $\mathcal{N}_{\text{active}} = \bigcup_{j=1}^{T} \mathcal{N}^j_{\text{active}}$, i.e., the $\{\mathcal{P}, \mathcal{P}_{\text{active}}, \mathcal{N}^0_{\text{active}}, \mathcal{N}_{\text{active}}\}$. More details will be given in Subsections 3.2–3.4.

3.2 Decision boundary detection

In our proposed ALIS framework, any decision boundary detection method (such as SVM and its extensions) can be used to separate the majority class and minority class. In this study, we apply SVM and large distribution machine (LDM) [31] as the classifier. LDM tries to achieve a better generalization performance by optimizing the margin distribution by the first- (margin mean) and second-order (margin variance) statistics, which are the mean of each component and the correlations between components respectively [32]. More detailly, f is regarded as a linear model:

$$f(\boldsymbol{x}) = \omega^{\mathrm{T}}\boldsymbol{x}, \quad \omega = (\omega_0, \ldots, \omega_m), \tag{1}$$

where ω is a linear predictor and the margin for $\boldsymbol{x}_i$ is defined as

$$\gamma_i = y_i f(\boldsymbol{x}_i), \quad \forall i = 1, \ldots, m. \tag{2}$$

Then, the margin mean $\bar{\gamma}$ and variance $\hat{\gamma}$ are calculated via

$$\bar{\gamma} = \frac{1}{n}\sum_{i=1}^{n} y_i f(\boldsymbol{x}_i), \tag{3}$$

$$\hat{\gamma} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(y_i f(\boldsymbol{x}_i) - y_j f(\boldsymbol{x}_j))^2. \tag{4}$$

Then, similar to soft-margin SVM, the soft-margin LDM leads to

$$\min_{\omega,\xi} \frac{1}{2}\omega^{\mathrm{T}}\omega + \lambda_1\hat{\gamma} - \lambda_2\bar{\gamma} + C\sum_{i=1}^{m}\xi_i \tag{5}$$

$$\text{s.t.}\ \ y_i\omega^{\mathrm{T}}\phi(x_i) \geqslant 1-\xi_i, \tag{6}$$

$$\xi_i \geqslant 0,\ i=1,\ldots,m, \tag{7}$$

where λ_1 and λ_2 are the parameters for trading-off the margin variance and the margin mean, respectively. SVM is a special case of LDM with $\lambda_1=\lambda_2=0$.

3.3 Important undersampling

Except for the initial step where ALIS randomly selects negative instances as $\mathcal{N}^0_{\text{active}}$ for each iteration, ALIS tries to select the important majority instances around the boundary. To implement this, it considers the structure information of these instances such as their first- and second- statistics. More specifically, for a candidate set, its margin distribution is captured by

$$\text{ratio} = \frac{\bar{\gamma}}{\hat{\gamma}}, \tag{8}$$

where $\bar{\gamma}$ and $\hat{\gamma}$ denote the average and variance of the margins of instances from the candidate set to the current decision boundary respectively.

In each active learning stage, it will select several important majority-class instances to form N^j_{active}. In practice, m instances (m is set to be 3 in experiments) closest to the current boundary are used as the initial candidate set $N^{j^0}_{\text{active}}$, and its margin distribution ratio can be calculated via (8), denoted as ratio^{j^0}. Next, we construct N^j_{active} in an iterative way. In each iteration, t extra instances (following the active learning strategy, t is set to be 2 in experiments) closest to the boundary are selected from pool and added to $N^{j^k}_{\text{active}}$. Similarly, we can compute its margin distribution ratio^{j^k}. With the increasing of iteration, more majority-class instances will be added. To make sure the selected instances have better margin distribution, this process will stop when the corresponding ratio^{j^k} increases. Among them, small ratio^{j^k} indicates small distance average and large distance variance. The former is helpful to select uncertain instances because they are close to the boundary. The latter is helpful to select diverse instances because they are scatter to each other. As pointed out in [33], uncertainty and diversity are two main requirements of active learning when selecting instances from pool.

Thus, if $\text{ratio}^{j^{k+1}} > \text{ratio}^{j^k}$, the procedure of selecting informative negative points comes to an end with $N^j_{\text{active}} = N^{j^k}_{\text{active}}$. By capturing margin distribution of selected points, ALIS ensures the points used for adjusting model are informative and representative.

3.4 Important oversampling

To avoid imbalance ratio becomes larger, ALIS applies important oversampling which generates new boundary minority points. It first selects informative borderline instances (sampling seeds) from the minority class. Then it designs an appropriate generator to create new minority points.

Proper sampling seeds are usually found via the k-nearest neighbor-based approach because the boundary minority class instances are more likely to have majority class instances among its nearest neighbors. Let $\text{NN}(\boldsymbol{x}_i)$ indicate the neighbors set for the instance $\boldsymbol{x}_i$ excluding itself. The informative set of positive class is denoted as $\mathcal{P}^{\text{info}}$, i.e.,

$$\mathcal{P}^{\text{info}} = \{\boldsymbol{p}_i | \boldsymbol{p}_i \in \mathcal{P}\ \cap\ \boldsymbol{x}_i \in \text{NN}(\boldsymbol{p}_i)\ \cap\ \boldsymbol{x}_i \in \mathcal{N}\}. \tag{9}$$

Let k_{i-} indicate the amount of majority points from $\text{NN}(\boldsymbol{p}_i)$, i.e.,

$$k_{i-} = |\{\boldsymbol{x}_i | \boldsymbol{p}_i \in \mathcal{P}^{\text{info}}\ \cap\ \boldsymbol{x}_i \in \text{NN}(\boldsymbol{p}_i)\ \cap\ \boldsymbol{x}_i \in \mathcal{N}\}|. \tag{10}$$

Thus, for each point $\boldsymbol{x}_i \in \mathcal{P}^{\text{info}}$, a weighted value ξ_i is assigned by the distribution of its k nearest neighbors set:

$$\xi_i = \frac{k_{i-}}{k}. \tag{11}$$

After fixing the sampling seeds and their importance, Kernel density function is adopted to capture boundary points' distribution and generate new boundary points.

The density distribution of informative minority class instances ($\boldsymbol{x}_i \in \mathcal{P}^{\text{info}}$) can be written by

$$G_i(\boldsymbol{x}) = \frac{1}{(\sqrt{2\pi})^m h_i} \exp\Big(-\frac{1}{2}\frac{(\boldsymbol{x}-\boldsymbol{x}_i)'(\boldsymbol{x}-\boldsymbol{x}_i)}{h_i}\Big), \tag{12}$$

where $h_i = \min \text{dis}(\boldsymbol{x}_i, \text{NN}(\boldsymbol{x}_i))$, indicating the bandwith, and dis is a distance function. Then, the estimated probability density function $p(\boldsymbol{x})$ based on $\mathcal{P}^{\text{info}}$ can be written as

$$p(\boldsymbol{x}) = \sum_{i}^{n_{ib}^{+}} \xi_i G_i(\boldsymbol{x}), \tag{13}$$

where n_{ib}^{+} is the number of instances in $\mathcal{P}^{\text{info}}$, i.e., $n_{ib}^{+} = |\mathcal{P}^{\text{info}}|$. Then the PDF estimation from informative borderline set can be interpreted as Gaussian mixture model with n_{ib}^{+} components and ξ_i is the mixture weight.

The oversampling strategy is similar with KernelADASYN [14]. But, they are slightly different on calculating ξ_i and h_i. When calculating ξ_i, for each selected minority-class seed, KernelADASYN finds its k-nearest neighbor instances from the whole training set including all minority-class and majority-class instances. ALIS is an iterative learning process. In each step, ALIS finds the nearest neighbors of each seed from the current training set which covers all given minority-class instances, the selected important majority-class instances and the generated informative minority-class instances. The number of generated instances depends on the number of selected important majority-class instances, which make sure the training data balance in each active learning stage. Another point is that KernelADASYN predefines the bandwidth h_i, and each component in (13) is assumed having the same variance, i.e., $h = h_1 = \cdots = h_i = \cdots = h_{n_{ib}^{+}}$. Such assumption will be violated in the complicated situation because some components may be dense which will prefer to small variance, and some components are sparse which prefer to large variance. Thus, h_i is self-tuned in ALIS according to its nearest neighbors. ALIS tries to select boundary instances as sampling seeds because the examples on the borderline and the ones nearby are more uncertain, thus more apt to be misclassified than the ones far from the borderline. Then, ALIS adaptively sets the importance of each seed and approximates the probability density function around it by (11)–(13). In this case, each seed is considered as a center and the corresponding bandwidth is adaptively determined, which has ability to introduce data perturbation for robust learning. Following this distribution, we can make sure the synthetic samples are more diverse than random replication and linear interpolation.

To give a more comprehensive understanding of proposed framework ALIS, the algorithm flowcharts of the proposed two important sampling strategies are provided in Algorithms 1 and 2, respectively. In addition, the graphical representation of ALIS is shown in Appendix A.

Algorithm 1 Important undersampling algorithm

```
Input: Classifier$^j$, pool negative dataset $\mathcal{N}_{\text{pool}}$, batchsize.
Output: actively selected negative dataset $\mathcal{N}^j_{\text{active}}$.
1: Initialize times = 0; ratio$_1$ = 1; ratio$_2$ = 0;
   $\mathcal{N}'_{\text{pool}}$: order $\mathcal{N}_{\text{pool}}$ by the according distance between instances and decision boundary of Classifier$^j$;
2: while ratio$_2$ < ratio$_1$ do
3:     times = times + 1;
4:     $\mathcal{N}_1$= top $\sharp$(times $\times$ batchsize) instances in $\mathcal{N}'_{\text{pool}}$;
5:     $\mathcal{N}_2$= top $\sharp$((times + 1) $\times$ batchsize) instances in $\mathcal{N}'_{\text{pool}}$;
6:     Calculate ratio$_1$ and ratio$_2$ of $\mathcal{N}_1$ and $\mathcal{N}_2$ according to (8) respectively;
7: end while
8: $\mathcal{N}^j_{\text{active}} = \mathcal{N}_1$.
```

Table 2 Description of the datasets

Dataset	n	m	n^-	n^+	ratio ($\frac{n^-}{n^+}$)
haberman	306	3	225	81	2.8
libra	360	90	288	72	4
glass6	214	9	185	29	6.38
ecoli3	336	7	301	35	8.6
yeast0256vs3789	1004	8	905	99	9.14
Satimage	6435	36	5809	626	9.27
balance	625	4	576	49	11.8
shuttlec0vsc4	1829	9	1706	123	13.87
Letter-a	20000	16	19211	789	24.34
yeast4	1484	8	1433	51	28.1
yeast6	1484	8	1449	35	41.4
abalone19	4174	7	4142	32	129.44

Algorithm 2 Important oversampling algorithm

```
Input: $\mathcal{P}_{\text{active}}$, $\mathcal{N}_{\text{active}}$, $k$.
Output: synthetic minority dataset $\mathcal{P}^j_{\text{active}}$.
1: Identify informative minority-class set $\mathcal{P}^{\text{info}}$ via (9);
2: for $\boldsymbol{x}_i \in \mathcal{P}^{\text{info}}$ do
3:     Set the mixture weight $\xi_i$ via (11);
4:     Set the bandwidth $h_i = \min \text{dis}(\boldsymbol{x}_i, \text{NN}(\boldsymbol{x}_i))$;
5: end for
6: Generate the synthetic minority instances by GMM estimation via (13).
```

4 Experiments and results

4.1 Data sets

The experimental binary-class data sets are from UCI data repository [34] and the KEEL data repository [35], which are commonly studied in imbalance classification [11, 13, 16, 36, 37]. Details of these datasets are shown in Table 2. n is the number of total instances, m is the number of features describing the data, n^+ and n^- are the numbers of minority and majority instances, respectively. Obviously, these datasets have a wide range of imbalance ratio (from 2.8 to 129.44), where larger ratio will make the learning problem more difficult. In addition, it is worth to point out that datasets with imbalance ratio over 10 can be regarded as the highly imbalanced datasets [38].

4.2 Methodology

To verify the effectiveness of ALIS, we compare it with five popular methods which deal with imbalance classification from different points of view. RoSR-L [11] is cost sensitive-based methods. KernelADASYN [14] is an oversampling method. AdaS [4] takes advantage of ensemble learning strategy which involves both undersampling and oversampling. AQM [30] and BorS [26] make use of active undersampling strategy. The proposed ALIS integrates undersampling and oversampling with the aid of active learning strategy.

In experiment, when detecting the decision boundary, SVM and LDM are used. The trade-off parameters λ_1 and λ_2 in LDM and cost parameter C in both LDM and SVM are tuned. For important oversampling, number of nearest neighbors k is tuned from the candidate set $\{5, 8\}$. All the best parameters are selected via 5-fold cross validation from the training data at the first random data partition as CS-LDM [9] does. Because there are several parameters, in experiments, nested cross validation is adopted for model evaluation; i.e., 5-fold cross-validation is used for both validation set and testing set. Specifically, for each dataset, it is randomly divided into five equal-size folds. One by one, a fold is taken as testing. Then, one by one, one of the left folds is used as validation and the other three folds are for

Table 3 Comparison of ALIS and five baselines in terms of recall for minority class on twelve real-world datasets

Dataset	RoSR-LL	KernelADASYN	AdaS	AQM	BorS	ALIS$_{\rm SVM}$	ALIS$_{\rm LDM}$
haberman	0.5676 ± 0.05	0.4088 ± 0.10	0.4706 ± 0.10	0.7184 ± 0.14	0.0485 ± 0.07	0.7294 ± 0.06	**0.7676 ± 0.07**
libra	0.7933 ± 0.06	0.0971 ± 0.06	0.6124 ± 0.20	0.3276 ± 0.29	0.3238 ± 0.19	0.5724 ± 0.16	**0.8067 ± 0.14**
glass6	**0.9754 ± 0.01**	0.8667 ± 0.22	0.8667 ± 0.22	0.9333 ± 0.09	0.8000 ± 0.30	0.9333 ± 0.09	**0.9668 ± 0.06**
ecoli3	**0.9714 ± 0.06**	0.5143 ± 0.16	0.9429 ± 0.08	**0.9714 ± 0.06**	0.2000 ± 0.19	**0.9714 ± 0.06**	**0.9714 ± 0.06**
yeast0256vs3789	0.7479 ± 0.08	0.5958 ± 0.17	0.6763 ± 0.12	0.8479 ± 0.05	0.1816 ± 0.16	0.6763 ± 0.15	**0.9103 ± 0.07**
Satimage	0.6917 0.04	0.0000 0.00	0.0000 0.00	**0.9856 0.01**	0.0000 0.00	0.2555 ± 0.03	**0.9170 ± 0.04**
balance	0.1400 ± 0.17	0.0000 ± 0.00	0.2822 ± 0.14	0.5556 ± 0.14	0.0000 ± 0.00	**0.7600 ± 0.15**	0.6600 ± 0.43
shuttlec0vsc4	**1.0000 ± 0.00**	0.6396 ± 0.13	0.1890 ± 0.13	0.4505 ± 0.20	0.0868 ± 0.12	0.4637 ± 0.16	**1.0000 ± 0.00**
Letter-a	0.9252 ± 0.02	0.9354 ± 0.02	0.9506 ± 0.01	0.9924 ± 0.01	0.9772 ± 0.01	**0.9937 ± 0.01**	0.9454 ± 0.01
yeast4	0.8600 ± 0.11	0.3327 ± 0.05	0.0982 ± 0.10	**0.9400 ± 0.09**	0.0000 ± 0.00	0.8218 ± 0.09	**0.9400 ± 0.09**
yeast6	0.8857 ± 0.12	0.6000 ± 0.16	0.0286 ± 0.06	0.8857 ± 0.12	0.0571 ± 0.08	0.9143 ± 0.13	**0.9435 ± 0.08**
abalone19	0.4429± 0.15	0.0000± 0.00	0.0000± 0.00	0.6714± 0.32	0.0000± 0.00	**0.7286 ± 0.25**	0.6286 ± 0.21
Winning times	3	0	0	3	0	4	**9**

training until all possible combinations have been evaluated. Finally, the training set is used to build the learning model, the validation set is used for selecting the proper parameters, and then the testing set is used to evaluate the model with the best parameters. The averaged results and the corresponding statistics are recorded. To validate the imbalance classification performance, the mean and standard deviation of four commonly used in imbalanced learning metrics are adopted, i.e., recall of minority class, precision of majority class, macro-averaged F-measure (F$_{\rm macro}$) and AUC. The average results with standard deviation are recorded. For imbalance learning, because the positive class is of more interest, it is expected that out of all the positive observations more of them are predicted as positive. In the meanwhile, with high confidence of the probability that a predicted negative instance indeed belongs to the negative class. So, a good model needs to return a higher recall on the positive class and higher precision on the negative class [39]. Besides, (F$_{\rm macro}$) and AUC are also used to give a fair evaluation of these methods.

4.3 Comparison with baselines

Tables 3–6 show the results in terms of precision of majority class, recall of minority class, F$_{\rm macro}$ and AUC. Moreover, we count the total number of winning times for each method across different evaluation metrics in each table. For each dataset, bold fonts and underline indicate the top two results respectively.

4.3.1 *Performance on the minority class*

Table 3 shows the performance of six methods on the minority class and high recall ensures that an algorithm has a good prediction performance on the minority class.

RoSR-LL and AQM perform better among five comparative methods while the others fail to recognize the minority class instances in certain data sets. Except the dataset Letter-a, KernelADASYN, AdaS and BorS degrade when the dataset with higher imbalance ratio. In real-world application, the minority class is usually of more interest. Recall for minority class shows the fraction of the total amount of relevant instances that actually belong to minority class. Here, zero recall (on minority class) indicates that KernelADASYN, AdaS and BorS incorrectly assign all minority-class instances to majority class for datasets with higher imbalanced ratio. The main reasons are as follows. KernelADASYN makes up for the imbalance problem via generating new positive points. When the imbalance ratio is high, however, insufficient boundary information cannot guarantee a good choice for sampling seeds. In this case, we cannot make sure to generate informative minority-class instances, which will degrade the classification performance. The main idea of AdaS is to generate several subsets of majority-class instances and oversample minority-class instances for each subset. One weak classifier will be built for each subset, however, it is hard to build a high-quality weak classifier for each subset owing to the limitation of training data. BorS first constructs an initial SVM classifier under a subset of the original imbalanced data and then expends the training set by adding the borderline instances. Owing to the imbalance between the majority class and minority class, the majority-class instances will be selected with high probability, which will deteriorate the subsequent learning process. In ALIS, though we do not have

Table 4 Comparison of ALIS and five baselines in terms of precision for majority class on twelve real-world datasets

Dataset	RoSR-LL	KernelADASYN	AdaS	AQM	BorS	ALIS_{SVM}	ALIS_{LDM}
haberman	0.8185 ± 0.02	0.7879 ± 0.04	0.7812 ± 0.04	0.7614 ± 0.12	0.7311 ± 0.01	0.8190 ± 0.05	**0.8856 ± 0.04**
libra	0.9349 ± 0.02	0.8145 ± 0.01	0.8932 ± 0.03	0.8286 ± 0.05	0.8560 ± 0.04	0.8555 ± 0.06	**0.9457 ± 0.03**
glass6	**0.9963 ± 0.00**	0.9797 ± 0.03	0.9796 ± 0.03	0.9884 ± 0.02	0.9702 ± 0.04	0.9895 ± 0.01	**0.9947 ± 0.01**
ecoli3	0.9918 ± 0.01	0.9446 ± 0.02	0.9919 ± 0.01	0.9960 ± 0.01	0.9148 ± 0.02	0.9959 ± 0.01	**0.9964 ± 0.01**
yeast0256vs3789	0.9674 ± 0.01	0.9559 ± 0.02	0.9633 ± 0.01	0.9591 ± 0.02	0.9174 ± 0.01	0.9638 ± 0.02	**0.9764 ± 0.01**
Satimage	0.9558 ± 0.00	0.9027 ± 0.00	0.9027 ± 0.00	0.0000 ± 0.00	0.9027 ± 0.00	0.9252 ± 0.00	**0.9831 ± 0.01**
balance	0.9186 ± 0.01	0.8980 ± 0.01	0.9295 ± 0.01	0.9463 ± 0.02	0.9213 ± 0.00	0.9663 ± 0.02	**0.9700 ± 0.03**
shuttlec0vsc4	**1.0000 ± 0.00**	0.9857 ± 0.01	0.9691 ± 0.01	0.9786 ± 0.01	0.9653 ± 0.00	0.9791 ± 0.01	**1.0000 ± 0.00**
Letter-a	0.9965 ± 0.00	0.9973 ± 0.00	0.9980 ± 0.00	**0.9997 ± 0.00**	0.9991 ± 0.00	**0.9997 ± 0.00**	0.9979 ± 0.00
yeast4	0.9941 ± 0.00	0.9763 ± 0.00	0.9688 ± 0.00	0.9957 ± 0.01	0.9656 ± 0.00	0.9924 ± 0.00	**0.9963 ± 0.00**
yeast6	0.9967 ± 0.00	0.9903 ± 0.00	0.9770 ± 0.00	0.9958 ± 0.00	0.9777 ± 0.00	**0.9974 ± 0.00**	0.9973 ± 0.00
abalone19	0.9949 ± 0.00	0.9922 ± 0.00	0.9923 ± 0.00	0.9933 ± 0.01	0.9923 ± 0.00	0.9964 ± 0.00	**0.9965 ± 0.00**
Winning times	2	0	0	1	0	2	**10**

access to enough positive instances, important positive points are generated during training process. By creating a more specific boundary, prediction for the minority class instances is promoted. These results further demonstrate that the proposed informative sampling is helpful to handle imbalanced data.

4.3.2 *Performance on the majority class*

Table 4 shows the performance of six methods on the majority class and higher precision ensures that an algorithm is more reliable if it predicts a candidate as the majority class. From the table, all the comparative methods have displayed a rather excellent performance on the majority. But the proposed method ALIS still improves the precision for the majority class for all the datasets. Though the imbalance ratio covers a wide range, ALIS performs quite well.

In important undersampling module in ALIS framework, negative instances will be discarded if the classification algorithm has a stable and good prediction accuracy on those instances. So, for ALIS, if the user-defined criterion is satisfied, the learner almost reaches an ideal state. ALIS picks up important instances and then gives away those redundant negative samples after testing them. More importantly, by comparing baselines and ALIS from Tables 3 and 4, ALIS has ability to improve the prediction performance for the minority class without losing confidence of predictive results on the majority class.

4.3.3 *Overall performance*

The algorithm performance on those two classes has been evaluated separately. To assess the overall classification performance, F_{macro} and AUC are adopted here and they are good indicators of classification performance in imbalance learning. As shown in Tables 5 and 6, the proposed ALIS achieves competitive even better classification performance in terms of AUC. Because AdaS uses majority vote to integrate weaker classifiers' outputs, it is meaningless to integrate scores given by all the weaker classifiers; and the model for KernelADASYN does not get such a probability score, so they are not included in Table 6.

ALIS uses borderline negative instances to find more precise boundary positive instances to generate new informative points and in return, positive points help adjust the boundary and more representative negative points are found. Because those two processes have an influence on each other, ALIS chooses those informative instances for training. Finally, ALIS yields an excellent performance on the majority class and improves detection rate of the minority class. Additionally, by counting the number of winning times, as shown in the last row of Tables 3–6, we can observe that ALIS outperforms all baselines in terms of recall for the minority class, precision for the majority class and AUC.

4.3.4 *Statistical analysis and run-time analysis*

To statistically analyze the prediction results, we adopt ANOVA to test the significance of differences between multiple classifiers' performance. Smaller ANOVA value indicates the difference among methods is more significant. As shown in Table 7, it shows that all means of all methods are significantly different.

Table 5 Comparing ALIS with five baselines in terms of F_{macro} on twelve real-world datasets

Dataset	RoSR-LL	KernelADASYN	AdaS	AQM	BorS	$\mathrm{ALIS}_{\mathrm{SVM}}$	$\mathrm{ALIS}_{\mathrm{LDM}}$
haberman	0.6151 ± 0.03	0.5997 ± 0.07	0.5621 ± 0.05	0.4299 ± 0.09	0.4401 ± 0.02	0.5108 ± 0.04	**0.7114 ± 0.05**
libra	0.7071 ± 0.06	0.5285 ± 0.05	0.6517 ± 0.08	0.5282 ± 0.08	0.6910 ± 0.14	0.5634 ± 0.11	**0.7520 ± 0.11**
glass6	0.7367 ± 0.03	0.9281 ± 0.08	0.9159 ± 0.07	0.8554 ± 0.07	0.9027 ± 0.11	0.8763 ± 0.10	**0.9368 ± 0.06**
ecoli3	0.6894 ± 0.03	0.7505 ± 0.08	0.7015 ± 0.03	0.6922 ± 0.05	0.6151 ± 0.13	0.6904 ± 0.04	**0.7554 ± 0.05**
yeast0256vs3789	0.6600 ± 0.02	0.7621 ± 0.05	0.7555 ± 0.04	0.4089 ± 0.09	0.6087 ± 0.09	**0.7722 ± 0.06**	0.6400 ± 0.06
Satimage	0.5710 ± 0.01	0.4744 ± 0.00	0.4744 ± 0.00	0.0875 ± 0.00	0.4744 ± 0.00	**0.6709 ± 0.01**	0.4825 ± 0.02
balance	0.4754 ± 0.06	0.4084 ± 0.01	0.5073 ± 0.04	0.5012 ± 0.09	0.4787 ± 0.00	0.4950 ± 0.06	**0.5193 ± 0.06**
shuttlec0vsc4	0.8503 ± 0.02	0.7897 ± 0.04	0.6289 ± 0.09	0.7316 ± 0.08	0.5516 ± 0.07	0.7344 ± 0.07	**0.8506 ± 0.03**
Letter-a	0.6625 ± 0.01	0.9488 ± 0.01	0.9858 ± 0.00	0.9339 ± 0.02	**0.9904 ± 0.00**	0.9864 ± 0.00	0.7222 ± 0.02
yeast4	0.5767 ± 0.02	**0.6584 ± 0.03**	0.5620 ± 0.07	0.2302 ± 0.15	0.4913 ± 0.00	0.5620 ± 0.03	0.4980 ± 0.01
yeast6	0.5549 ± 0.01	**0.7436 ± 0.06**	0.5161 ± 0.05	0.3831 ± 0.14	0.5390 ± 0.06	0.5321 ± 0.02	0.6413 ± 0.07
abalone19	0.4803 ± 0.01	0.4950 ± 0.00	0.4981 ± 0.00	0.3207 ± 0.12	0.4981 ± 0.00	0.3731 ± 0.02	**0.5061 ± 0.04**
Winning times	0	2	0	0	1	2	**7**

Table 6 Comparing ALIS with three baselines in terms of AUC on twelve real-world datasets

Dataset	RoSR-LL	AQM	BorS	$\mathrm{ALIS}_{\mathrm{SVM}}$	$\mathrm{ALIS}_{\mathrm{LDM}}$
haberman	0.6718 ± 0.09	0.4837 ± 0.09	0.4894 ± 0.13	0.5665 ± 0.05	**0.7192 ± 0.09**
libra	0.5470 ± 0.07	0.2627 ± 0.12	0.2071 ± 0.04	0.6426± 0.12	**0.8313 ± 0.07**
glass6	0.8214 ± 0.2445	0.0387 ± 0.0721	0.0324 ± 0.0676	**0.9757 ± 0.05**	0.9676 ± 0.03
ecoli3	0.4065 ± 0.0647	0.1175 ± 0.0427	0.0600 ± 0.0478	0.9152 ± 0.04	**0.9289 ± 0.02**
yeast0256vs3789	0.6774 ± 0.0366	0.3108 ± 0.0937	0.1558 ± 0.0376	**0.8336 ± 0.05**	0.8320 ± 0.03
Satimage	0.1809 ± 0.03	**0.9990 ± 0.00**	0.2580 ± 0.34	0.9092 ± 0.01	0.7283 ± 0.02
balance	0.5822 ± 0.0640	0.2819 ± 0.1144	0.2395 ± 0.0331	**0.7056 ± 0.09**	0.5184 ± 0.06
shuttlec0vsc4	0.9053 ± 0.0851	0.0303 ± 0.0238	0.0687 ± 0.0315	0.9848 ± 0.00	**0.9858 ± 0.00**
Letter-a	0.7689 ± 0.04	0.0004 ± 0.00	0.0001 ± 0.00	**0.9999 ± 0.00**	0.9845± 0.00
yeast4	0.5844 ± 0.04	0.5133 ± 0.27	0.1934 ± 0.06	**0.8642 ± 0.05**	0.7472 ± 0.12
yeast6	0.6766 ± 0.12	0.2309 ± 0.11	0.0848 ± 0.06	**0.9179 ± 0.08**	0.7912 ± 0.05
abalone19	0.2764 ± 0.05	0.4679 ± 0.25	0.4057 ± 0.13	0.7121 ± 0.09	**0.7177 ± 0.11**
Winning times	0	1	0	**6**	5

Moreover, we also perform pairwise t-test between ALIS and all other baselines for comparisons of two classifiers over different data sets. ALIS wins the baseline if the t-test value is below 0.05. The times of ALIS winning are indicated by the molecule in bracket. For precision-majority and recall-minority, there are five baselines involved, and that for AUC is three, so the denominator is 5 and 3, respectively. As expected in Table 7, ALIS makes a promising difference in comparison with other methods.

ALIS is a learning framework including informative sampling (undersampling and oversampling) and classifier (e.g., SVM, LDM and etc.) training. Table 8 lists the averaged running time (in seconds for each iteration) of three parts (undersampling (ALIS-Under), oversampling (ALIS-Over), and classifier building (ALIS-Classifier)). In experiments, all methods are implemented on the server with Intel Core i7 Processors and 8 G RAM. As expected, oversampling takes a lot because it has to generate synthetic instances in the original feature space. Fortunately, ALIS is efficient and almost linearly scale to data size.

4.4 Investigating ALIS

ALIS consists of three components, i.e., important undersampling, decision boundary detection and important oversampling. Thus, to study the effect of each component individually, we replace the proposed method for each part with the existing one and check the results. More details can be found in Appendix B. As expected, ALIS outperforms over any other situation which replaces one component with the existing method. This result further confirms that the proposed framework benefits from each component.

Table 7 Analysis of variance (ANOVA) test and winning times of pairwise t-test (in bracket) between ALIS and the baseline on twelve real-world datasets

Metric	haberman	libra	glass6	ecoli3	yeast0256vs3789	Satimage
Precision-majority	4.79E−03 (4)	1.18E−06 (4)	0.64 (0)	3.86E−10 (2)	2.48E−05 (2)	1.36E−46 (5)
Recall-minority	9.36E−10 (4)	2.38E−06 (2)	0.62 (0)	1.21E−10 (2)	7.28E−09 (4)	5.19E−31 (5)
F_{macro}	2.79E−07 (5)	0.0020 (2)	0.0011 (2)	5.44E−02 (4)	1.27E−08 (3)	2.00E−36 (5)
AUC	0.033 (2)	3.45E−09 (3)	1.79E−09 (2)	5.45E−15 (3)	7.83E−09 (3)	2.59E−06 (3)
Metric	**balance**	**shuttlec0vsc4**	**Letter-a**	**yeast4**	**yeast6**	**abalone19**
Precision-majority	8.37E−05 (4)	7.20E−12 (4)	8.70E−09 (4)	4.36E−13 (3)	1.30E−10 (3)	0.15 (3)
Recall-minority	4.53E−05 (3)	4.14E−12 (4)	1.82E−08 (4)	6.85E−17 (3)	5.23E−14 (3)	1.32E−07 (3)
F_{macro}	0.0249 (1)	1.40E−07 (4)	1.79E−27 (4)	2.74E−08 (2)	3.72E−06 (2)	0.0428 (0)
AUC	2.01E−06 (2)	5.15E−17 (2)	8.43E−22 (3)	0.8846 (1)	4.07E−05 (2)	4.48E−06 (2)

Table 8 Running time of each part in ALIS framework (s)

Method	haberman	libra	glass6	ecoli3	yeast0256vs3789	Satimage	balance	shuttlec0vsc4	Letter-a	yeast4	yeast6	abalone19
ALIS-Under	0.0003	0.0150	0.0005	0.0006	0.0006	0.0079	0.0002	0.0005	0.0593	0.0005	0.0006	0.0010
ALIS-Over	0.0153	0.1720	0.0126	0.0138	0.0925	0.1410	0.0373	0.0716	0.1870	0.0608	0.0344	0.1100
ALIS-Classifier	0.0002	0.0121	0.0000	0.0231	0.0123	0.0230	0.0001	0.0009	0.0920	0.0094	0.0131	0.0150

As we known, a multi-class classification problem can be decomposed into several binary classification problems, such as multi-class SVM. Thus, with the aid of two more multi-class datasets, we extend ALIS for multi-class classification task. More details can be found in Appendix C. The overall performance in terms of each metric shows that ALIS scales to multi-class problems via decomposition into binary-class classification problems.

5 Conclusion

Imbalance classification has drawn a wide attention. Stressed by two key points of imbalance classification, we propose an important sampling based ALIS in this paper. For undersampling, ALIS designs an active selection criterion which captures distribution of the selected negative instances. For oversampling, ALIS first selects boundary positive points. Then, after capturing the distribution of those informative points, new informative boundary points are generated. Experimental results with several performance evaluation metrics show the effectiveness of ALIS. The tested data sets have a large range of imbalance ratio (from 2.8 to 129.44), but ALIS performs quiet well on both precision for the negative class and recall for the positive class. In the meanwhile, it also has a good performance in terms of AUC. One more thing is that to directly extend ALIS for multi-class data by integrating the classifier such as neural network would be an interesting topic.

Acknowledgements This work was supported in part by National Natural Science Foundation of China (Grant Nos. 61822601, 61773050, 61632004, 61972132), Beijing Natural Science Foundation (Grant No. Z180006), National Key Research and Development Program (Grant No. 2017YFC1703506), Fundamental Research Funds for the Central Universities (Grant Nos. 2019JBZ110, 2019YJS040), Youth Foundation of Hebei Education Department (Grant No. QN2018084), Science and Technology Foundation of Hebei Agricultural University (Grant No. LG201804), and Research Project for Self-cultivating Talents of Hebei Agricultural University (Grant No. PY201810).

Supporting information Appendixes A–C. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Xu C, Tao D, Xu C. Robust extreme multi-label learning. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. 1275–1284

2 Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, 2017. 2980–2988

3 Batuwita R, Palade V. Efficient resampling methods for training support vector machines with imbalanced datasets. In: Proceedings of the International Joint Conference on Neural Networks, 2010. 1–8

4 Peng Y. Adaptive sampling with optimal cost for class-imbalance learning. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence, 2015. 2921–2927

5 Attenberg J, Ertekin S. Class imbalance and active learning. In: Imbalanced Learning: Foundations, Algorithms, and Applications. Piscataway: Wiley-IEEE Press, 2013. 101–149

6 Guo J, Wan X, Lin H, et al. An active learning method based on mistake sampling for large scale imbalanced classification. In: Proceedings of International Conference on Service Systems and Service Management, 2017. 1–6

7 Stefanowski J. Dealing with data difficulty factors while learning from imbalanced data. In: Challenges in Computational Statistics and Data Mining. Berlin: Springer, 2016. 333–363

8 Alejo R, Valdovinos R M, García V, et al. A hybrid method to face class overlap and class imbalance on neural networks and multi-class scenarios. Pattern Recogn Lett, 2013, 34: 380–388

9 Cheng F, Zhang J, Wen C. Cost-sensitive large margin distribution machine for classification of imbalanced data. Pattern Recogn Lett, 2016, 80: 107–112

10 Chung Y A, Lin H T, Yang S W. Cost-aware pre-training for multiclass cost-sensitive deep learning. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, 2016. 1411–1417

11 Ren Y, Zhao P, Sheng Y, et al. Robust softmax regression for multi-class classification with self-paced learning. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017. 2641–2647

12 Chawla N V, Bowyer K W, Hall L O, et al. SMOTE: synthetic minority over-sampling technique. J Artif Intell Res, 2002, 16: 321–357

13 Han H, Wang W Y, Mao B H. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In: Proceedings of the 2005 International Conference on Advances in Intelligent Computing. Berlin: Springer, 2005. 878–887

14 Tang B, He H. KernelADASYN: kernel based adaptive synthetic data generation for imbalanced learning. In: Proceedings of IEEE Congress on Evolutionary Computation, 2015. 664–671

15 Zhou C, Liu B, Wang S. Cmo-smote: misclassification cost minimization oriented synthetic minority oversampling technique for imbalanced learning. In: Proceedings of the 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2016. 353–358

16 Barua S, Islam M M, Yao X, et al. MWMOTE—majority weighted minority oversampling technique for imbalanced data set learning. IEEE Trans Knowl Data Eng, 2014, 26: 405–425

17 Yuan J, Li J, Zhang B. Learning concepts from large scale imbalanced data sets using support cluster machines. In: Proceedings of the 14th ACM International Conference on Multimedia, 2006. 441–450

18 He H B, Garcia E A. Learning from imbalanced data. IEEE Trans Knowl Data Eng, 2009, 21: 1263–1284

19 Tahir M A, Kittler J, Yan F. Inverse random under sampling for class imbalance problem and its application to multi-label classification. Pattern Recogn, 2012, 45: 3738–3750

20 Galar M, Fernández A, Barrenechea E, et al. EUSBoost: enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. Pattern Recogn, 2013, 46: 3460–3471

21 Thanathamathee P, Lursinsap C. Handling imbalanced data sets with synthetic boundary data generation using bootstrap re-sampling and AdaBoost techniques. Pattern Recogn Lett, 2013, 34: 1339–1347

22 Settles B. Active Learning Literature Survey. Technical Report. University of Wisconsin-Madison Department of Computer Sciences, 2009

23 Lughofer E, Weigl E, Heidl W, et al. Integrating new classes on the fly in evolving fuzzy classifier designs and their application in visual inspection. Appl Soft Comput, 2015, 35: 558–582

24 Weigl E, Heidl W, Lughofer E, et al. On improving performance of surface inspection systems by online active learning and flexible classifier updates. Machine Vision Appl, 2016, 27: 103–127

25 Pratama M, Dimla E, Lai C Y, et al. Metacognitive learning approach for online tool condition monitoring. J Intell Manuf, 2019, 30: 1717–1737

26 Ertekin S, Huang J, Bottou L, et al. Learning on the border: active learning in imbalanced data classification. In: Proceedings of the 16th ACM Conference on Information and Knowledge Management, 2007. 127–136

27 Batuwita R, Palade V. Class imbalance learning methods for support vector machines. In: Imbalanced Learning: Foundations, Algorithms, and Applications. Piscataway: Wiley-IEEE Press, 2013. 83

28 Oh S, Lee M S, Zhang B T. Ensemble learning with active example selection for imbalanced biomedical data classification. IEEE/ACM Trans Comput Biol Bioinf, 2011, 8: 316–325

29 Chen Y, Mani S. Active learning for unbalanced data in the challenge with multiple models and biasing. In: Proceedings

of Workshop on Active Learning and Experimental Design, 2011. 113–126

30 Zhang X, Yang T, Srinivasan P. Online asymmetric active learning with imbalanced data. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. 2055–2064

31 Zhang T, Zhou Z H. Large margin distribution machine. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014. 313–322

32 Roweis S. Boltzmann machines. Lecture notes, 1995. https://ftp.cs.nyu.edu/∼roweis/notes/boltz.pdf

33 Yang Y, Ma Z, Nie F, et al. Multi-class active learning by uncertainty sampling with diversity maximization. Int J Comput Vision, 2015, 113: 113–127

34 Asuncion A, Newman D. Uci machine learning repository, 2007. http://archive.ics.uci.edu/ml

35 Alcalá-Fdez J, Fernández A, Luengo J, et al. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J Multiple-Valued Logic Soft Comput, 2010, 17: 255–287

36 Sun Z, Song Q, Zhu X, et al. A novel ensemble method for classifying imbalanced data. Pattern Recogn, 2015, 48: 1623–1637

37 Yan Q, Xia S, Meng F. Optimizing cost-sensitive SVM for imbalanced data: connecting cluster to classification. 2017. ArXiv: 170201504

38 Wu F, Jing X Y, Shan S, et al. Multiset feature learning for highly imbalanced data classification. In: Prcoeedings of the 31st AAAI Conference on Artificial Intelligence, 2017

39 More A. Survey of resampling techniques for improving classification performance in unbalanced datasets. 2016. ArXiv: 160806048